Chapter 24. Homomorphism

Zhenjiang Hu, Wei Zhang

School of Computer Science
Peking University

December 7, 2022

Longest Even Segment Problem

Given is a predicate p and a sequence x. Required is an efficient algorithm for computing some longest segment of x, all of whose elements satisfy p.

$$lsp\ even\ [3,1,4,1,5,9,2,6,5] = [2,6]$$

A homomorphism from a monoid $(\alpha, \oplus, id_\oplus)$ to a monoid $(\beta, \otimes, id_\otimes)$ is a function h satisfying the two equations:

$$\begin{array}{lcl} h\ id_\oplus & = & id_\otimes \\ h\ (x \oplus y) & = & h\ x \otimes h\ y \end{array}$$

Homomorphisms on Non-Empty Lists

A homomorphism from a semi-group $(\alpha, \oplus)$ to a semi-group $(\beta, \otimes)$ is a function h satisfying the following equation:

$$h\ (x \oplus y) \quad = \quad h\ x \otimes h\ y$$

Map: a Homomorphism

$$f * [a_1, a_2, \ldots, a_n] = [f\ a_1, f\ a_2, \ldots, f\ a_n]$$

$$\begin{array}{lcl} f * [\,] & = & [\,] \\ f * [a] & = & [f\ a] \\ f * (x \mathbin{+\!\!+} y) & = & (f * x) \mathbin{+\!\!+} (f * y) \end{array}$$

Reduce: a Homomorphism

$$\oplus/[a_1, a_2, \ldots, a_n] = a_1 \oplus a_2 \oplus \cdots \oplus a_n$$

$$\begin{array}{lcl}
\oplus/[\,] & = & id_{\oplus} \\
\oplus/[a] & = & a \\
\oplus/(x \mathbin{+\!\!+} y) & = & (\oplus/x) \oplus (\oplus/y)
\end{array}$$

Lemma (Promotion)

h is a homomorphism from $(\alpha, \oplus, id_{\oplus})$ *to* $(\beta, \otimes, id_{\otimes})$ *if and only if the following holds.*

$$h \cdot \oplus/ = \otimes/ \cdot h*$$

Lemma (Promotion)

h is a homomorphism from $(\alpha, \oplus, id_\oplus)$ *to* $(\beta, \otimes, id_\otimes)$ *if and only if the following holds.*

$$h \cdot \oplus/ = \otimes/ \cdot h*$$

Proof Sketch.

- $\Leftarrow$: *simple.*
- $\Rightarrow$: *by induction.*

Lemma (Promotion)

h is a homomorphism from $(\alpha, \oplus, id_\oplus)$ to $(\beta, \otimes, id_\otimes)$ if and only if the following holds.

$$h \cdot \oplus/ = \otimes/ \cdot h*$$

Proof Sketch.

- $\Leftarrow$: *simple.*
- $\Rightarrow$: *by induction.*

So we have

$$\begin{array}{rcl} f* \cdot +\!\!+/ & = & +\!\!+/ \cdot f** \\ \oplus/ \cdot +\!\!+/ & = & \oplus/ \cdot (\oplus/)* \end{array}$$

Lemma (Identity)

$$id = +\!\!+ \;/\cdot [\cdot]*$$

Lemma (Identity)

$$id = +\!\!+ \,/ \cdot [\cdot]*$$

Lemma

h is a homomorphism from the list monoid if and only if there exist f and $\oplus$ such that

$$h = \oplus/ \cdot f*$$

Proof

$\Rightarrow$:

$$
\begin{array}{cl}
 & h \\
= & \quad \{ \text{ definition of } id \text{ } \} \\
 & h \cdot id \\
= & \quad \{ \text{ identity lemma } \} \\
 & h \cdot +\!\!\!+ / \cdot [\cdot]* \\
= & \quad \{ \text{ } h \text{ is a homomorphism } \} \\
 & \oplus / \cdot h * \cdot [\cdot]* \\
= & \quad \{ \text{ map distributivity } \} \\
 & \oplus / \cdot (h \cdot [\cdot])* \\
= & \quad \{ \text{ definition of } h \text{ on singleton } \} \\
 & \oplus / \cdot f*
\end{array}
$$

$\Leftarrow$: We reason that $h = \oplus/ \cdot f*$ is a homomorphism by calculating

$$
\begin{array}{ll}
 & h \cdot +\!\!+ / \\
= & \quad \{ \text{ given form for } h \ \} \\
 & \oplus/ \cdot f* \cdot +\!\!+ / \\
= & \quad \{ \text{ map and reduce promotion } \} \\
 & \oplus/ \cdot (\oplus/ \cdot f*)* \\
= & \quad \{ \text{ hypothesis } \} \\
 & \oplus/ \cdot h*
\end{array}
$$

Examples of Homomorphisms

- $\#$: compute the length of a list.

$$\# = +/ \cdot K_1 *$$

Examples of Homomorphisms

- $\#$: compute the length of a list.

$$\# = +/ \cdot K_1 *$$

- *reverse*: reverses the order of the elements in a list.

$$\mathit{reverse} = \tilde{+\!\!+} / \cdot [\cdot] *$$

Here, $x \tilde{\oplus} y = y \oplus x$.

- *sort*: reorders the elements of a list into ascending order.

$$sort = \curlywedge\!\!\curlywedge \; / \cdot [\cdot]*$$

Here, $\curlywedge\!\!\curlywedge$ (pronounced merge) is defined by the equations:

$$\begin{array}{lcll}
x \curlywedge\!\!\curlywedge [\,] & = & x & \\
[\,] \curlywedge\!\!\curlywedge y & = & y & \\
([a] +\!\!+ x) \curlywedge\!\!\curlywedge ([b] +\!\!+ y) & = & [a] +\!\!+ (x \curlywedge\!\!\curlywedge ([b] +\!\!+ y)), & \text{if } a \leq b \\
& = & [b] +\!\!+ (([a] +\!\!+ x) \curlywedge\!\!\curlywedge y), & \text{otherwise}
\end{array}$$

- *all p*: returns True if every element of the input list satisfies the predicate p.

$$all\ p = \wedge/ \cdot p*$$

- *all* p: returns True if every element of the input list satisfies the predicate p.

$$all\ p = \wedge/ \cdot p*$$

- *some* p: returns True if at least one element of the input list satisfies the predicate p.

$$some\ p = \vee/ \cdot p*$$

Homework BMF 2-1

1. Show that function *split* that splits a non-empty list into its last element and the remainder is a homomorphism.

$$split\ [1, 2, 3, 4] = ([1, 2, 3], 4)$$

2. Let $init = \pi_1 \cdot split$, where $\pi_1\ (a, b) = a$. Show that *init* is not a homomorphism.

All applied to

The operator ^{o} (pronounced all applied to) takes a sequence of functions and a value and returns the result of applying each function to the value.

$$[f, g, \ldots, h]^o a = [f\ a, g\ a, \ldots, h\ a]$$

Formally, we have

$$\begin{array}{rcl} [\,]^o\ a & = & [\,] \\ [f]^o\ a & = & [f\ a] \\ (fs +\!\!+\ gs)^o\ a & = & (fs^o\ a) +\!\!+\ (gs^o\ a) \end{array}$$

so, $(^o\ a)$ is a homomorphism.

Exercise: Show that $[\cdot] = [id]^o$.

Conditional Expressions

The conditional notation

$$h\ x \quad = \quad \textbf{if}\ p\ x\ \textbf{then}\ f\ x\ \textbf{else}\ g\ x$$

will be written by the McCarthy conditional form:

$$h = (p \to f, g)$$

The conditional notation

$$h\ x \quad = \quad \textbf{if}\ p\ x\ \textbf{then}\ f\ x\ \textbf{else}\ g\ x$$

will be written by the McCarthy conditional form:

$$h = (p \to f, g)$$

Laws on Conditional Forms

$$\begin{array}{lcl} h \cdot (p \to f, g) & = & (p \to h \cdot f, h \cdot g) \\ (p \to f, g) \cdot h & = & (p \cdot h \to f \cdot h, g \cdot h) \\ (p \to f, f) & = & f \end{array}$$

(Note: all functions are assumed to be total.)

Filter

The operator $\triangleleft$ (pronounced filter) takes a predicate p and a list x and returns the sublist of x consisting, in order, of all those elements of x that satisfy p.

$$p\triangleleft = +\!\!+ / \cdot (p \rightarrow [id]^o, [\,]^o)*$$

Filter

The operator $\triangleleft$ (pronounced filter) takes a predicate p and a list x and returns the sublist of x consisting, in order, of all those elements of x that satisfy p.

$$p\triangleleft = +\!\!+ \,/ \cdot (p \rightarrow [id]^o, [\,]^o)*$$

Exercise: Prove that the filter satisfies the filter promotion property:

$$(p\triangleleft) \cdot +\!\!+ \,/ = +\!\!+ \,/ \cdot (p\triangleleft)*$$

Filter

The operator $\triangleleft$ (pronounced filter) takes a predicate p and a list x and returns the sublist of x consisting, in order, of all those elements of x that satisfy p.

$$p\triangleleft = +\!\!+ / \cdot (p \rightarrow [id]^o, [\,]^o)*$$

Exercise: Prove that the filter satisfies the filter promotion property:

$$(p\triangleleft) \cdot +\!\!+ / = +\!\!+ / \cdot (p\triangleleft)*$$

Exercise: Prove that the filter satisfies the map-filter swap property:

$$(p\triangleleft) \cdot f* = f * \cdot (p \cdot f)\triangleleft$$

Cross-product

$X_{\oplus}$ is a binary operator that takes two lists x and y and returns a list of values of the form $a \oplus b$ for all a in x and b in y.

$$[a, b] X_{\oplus} [c, d, e] = [a \oplus c, b \oplus c, a \oplus d, b \oplus d, a \oplus e, b \oplus e]$$

Formally, we define $X_{\oplus}$ by three equations:

Cross-product

$X_{\oplus}$ is a binary operator that takes two lists x and y and returns a list of values of the form $a \oplus b$ for all a in x and b in y.

$$[a, b] X_{\oplus} [c, d, e] = [a \oplus c, b \oplus c, a \oplus d, b \oplus d, a \oplus e, b \oplus e]$$

Formally, we define $X_{\oplus}$ by three equations:

$$\begin{array}{lcl} x X_{\oplus} [\,] & = & [\,] \\ x X_{\oplus} [a] & = & (\oplus a) * x \\ x X_{\oplus} (y +\!\!+ z) & = & (x X_{\oplus} y) +\!\!+ (x X_{\oplus} z) \end{array}$$

Thus $x X_{\oplus}$ is a homomorphism.

Properties

$[\,]$ is the zero element of $X_{\oplus}$:

$$[\,]X_{\oplus}x \;=\; xX_{\oplus}[\,] \;=\; [\,]$$

We have cross promotion rules:

$$\begin{array}{rcl} f**\cdot X_{+\!\!+}/ & = & X_{+\!\!+}/\cdot f*** \\ (\oplus/)*\cdot X_{+\!\!+}/ & = & X_{\oplus}/\cdot(\oplus/)** \end{array}$$

Example Uses of Cross-product

- *cp*: takes a list of lists and returns a list of lists of elements, one from each component.

$$cp\ [[a, b], [c], [d, e]] = [[a, c, d], [b, c, d], [a, c, e], [b, c, e]]$$

Example Uses of Cross-product

- *cp*: takes a list of lists and returns a list of lists of elements, one from each component.

$$cp\ [[a, b], [c], [d, e]] = [[a, c, d], [b, c, d], [a, c, e], [b, c, e]]$$

$$cp = X_{++}/ \cdot ([id]^{o}*)*$$

- *subs*: computes all subsequences of a list.

$$subs\ [a, b, c] = [[], [a], [b], [a, b], [c], [a, c], [b, c], [a, b, c]]$$

- *subs*: computes all subsequences of a list.

$$subs\ [a, b, c] = [[], [a], [b], [a, b], [c], [a, c], [b, c], [a, b, c]]$$

$$subs = X_{++} / \cdot [[\,]^o, [id]^o]^o *$$

- *subs*: computes all subsequences of a list.

$$subs\ [a, b, c] = [[], [a], [b], [a, b], [c], [a, c], [b, c], [a, b, c]]$$

$$subs = X_{++} / \cdot [[\,]^o, [id]^o]^o *$$

Exercise: Define *subs* in terms of *cp*.

- $(all\ p)\triangleleft$:

$$(all\ even) \triangleleft [[1,3],[2,4],[1,2,3]] = [[2,4]]$$

- $(all\ p)\triangleleft$:

$$(all\ even) \triangleleft [[1,3],[2,4],[1,2,3]] = [[2,4]]$$

$$(all\ p)\triangleleft = +\!\!+ \ / \cdot (X_{+\!\!+} / \cdot (p \rightarrow [[id]^o]^o, [\,]^o)*)*$$

- $(all\ p)\triangleleft$:

$$(all\ even) \triangleleft [[1,3],[2,4],[1,2,3]] = [[2,4]]$$

$$(all\ p)\triangleleft = +\!\!+ \ / \cdot (X_{+\!\!+} / \cdot (p \to [[id]^o]^o, [\,]^o)*)*$$

Exercise: Compute the value of the expression $(all\ even) \triangleleft [[1,3],[2,4],[1,2,3]]$.

Selection Operators

Suppose f is a numeric valued function. We want to define an operator $\uparrow_f$ such that

1. $\uparrow_f$ is associative, commutative and idempotent;
2. $\uparrow_f$ is selective in that

$$x \uparrow_f y = x \quad \text{or} \quad x \uparrow_f y = y$$

3. $\uparrow_f$ is maximizing in that

$$f(x \uparrow_f y) = f\,x \uparrow f\,y$$

Suppose f is a numeric valued function. We want to define an operator $\uparrow_f$ such that

1. $\uparrow_f$ is associative, commutative and idempotent;
2. $\uparrow_f$ is selective in that

$$x \uparrow_f y = x \quad \text{or} \quad x \uparrow_f y = y$$

3. $\uparrow_f$ is maximizing in that

$$f(x \uparrow_f y) = f\,x \uparrow f\,y$$

Condition: f should be injective.

An Example: $\uparrow_{\#}$

But if f is not injective, then $x \uparrow_f y$ is not specified when $x \neq y$ but $f\,x = f\,y$.

$$[1,2] \uparrow_{\#} [3,4]$$

An Example: $\uparrow_{\#}$

But if f is not injective, then $x \uparrow_f y$ is not specified when $x \neq y$ but $f\,x = f\,y$.

$$[1,2] \uparrow_{\#} [3,4]$$

To solve this problem, we may *refine* f to an injective function f' such that

$$f\,x < f\,y \Rightarrow f'\,x < f'\,y.$$

But if f is not injective, then $x \uparrow_f y$ is not specified when $x \neq y$ but $f\,x = f\,y$.

$$[1,2] \uparrow_{\#} [3,4]$$

To solve this problem, we may *refine* f to an injective function f' such that

$$f\,x < f\,y \Rightarrow f'\,x < f'\,y.$$

So we may select the *lexicographically* least sequence as the value of $x \uparrow_{\#} y$ when $\#x = \#y$.

In this case, $+\!\!+$ distributes through $\uparrow_{\#}$:

$$\begin{array}{rcl} x +\!\!+ (y \uparrow_{\#} z) & = & (x +\!\!+ y) \uparrow_{\#} (x +\!\!+ z) \\ (x \uparrow_{\#} y) +\!\!+ z & = & (x +\!\!+ z) \uparrow_{\#} (y +\!\!+ z) \end{array}$$

That is,

$$\begin{array}{rcl} (x +\!\!+)\cdot \uparrow_{\#} / & = & \uparrow_{\#} / \cdot (x +\!\!+)* \\ (+\!\!+\ x)\cdot \uparrow_{\#} / & = & \uparrow_{\#} / \cdot (+\!\!+\ x) * . \end{array}$$

We assume $\omega = \uparrow_{\#} /[]$, satisfying $\#\omega = -\infty$. (ω is the zero element of $+\!\!+$)

A short calculation

Show that $\uparrow_{\#} / \cdot (\mathit{all}\ p) \triangleleft$ is a homomorphism.

A short calculation

Show that $\uparrow_{\#} / \cdot (all\ p)\triangleleft$ is a homomorphism.

$$
\begin{array}{ll}
 & \uparrow_{\#} / \cdot (all\ p)\triangleleft \\
= & \quad \{ \text{ definition before } \} \\
 & \uparrow_{\#} / \cdot +\!\!+ / \cdot (X_{+\!\!+} / \cdot (p \to [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ \text{ reduce promotion } \} \\
 & \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot X_{+\!\!+} / \cdot (p \to [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ +\!\!+ \text{ distributes over } \uparrow_{\#} \} \\
 & \uparrow_{\#} / \cdot (+\!\!+ / \cdot (\uparrow_{\#} /) * \cdot (p \to [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ \text{ many steps ... } \} \\
 & \uparrow_{\#} / \cdot (+\!\!+ / \cdot (p \to [id]^o, K_{\omega})*)*
\end{array}
$$

Existence of Homomorphism

Existence Lemma

The list function h is a homomorphism iff the implication

$$h\ v = h\ x \ \wedge \ h\ w = h\ y \ \Rightarrow \ h\ (v +\!\!+ w) = h\ (x +\!\!+ y)$$

holds for all lists v, w, x, y.

Existence Lemma

The list function h is a homomorphism iff the implication

$$h\ v = h\ x\ \wedge\ h\ w = h\ y\ \Rightarrow\ h\ (v +\!\!+\ w) = h\ (x +\!\!+\ y)$$

holds for all lists v, w, x, y.

Proof Sketch.

- $\Rightarrow$: obvious by assuming $h = \odot/ \cdot f*$.

Existence Lemma

The list function h is a homomorphism iff the implication

$$h\ v = h\ x\ \wedge\ h\ w = h\ y\ \Rightarrow\ h\ (v +\!\!+\ w) = h\ (x +\!\!+\ y)$$

holds for all lists v, w, x, y.

Proof Sketch.

- $\Rightarrow$: obvious by assuming $h = \odot/ \cdot f*$.
- $\Leftarrow$: Define $\odot$ by $t \odot u = h\ (g\ t +\!\!+\ g\ u)$.
  for some g such that $h = h \cdot g \cdot h$ (such a g exisits!). Thus

$$h\ (x +\!\!+\ y)\quad = \quad h\ x \odot h\ y.$$

Specification of the Problem

Recall the problem of computing the longest segment of a list, all of whose elements satisfied some given property p.

$$lsp = \uparrow_{\#} / \cdot (all\ p) \triangleleft \cdot segs$$

Specification of the Problem

Recall the problem of computing the longest segment of a list, all of whose elements satisfied some given property p.

$$lsp = \uparrow_{\#} / \cdot (all\ p) \triangleleft \cdot segs$$

Property: *lsp* is not a homomorphism.

Specification of the Problem

Recall the problem of computing the longest segment of a list, all of whose elements satisfied some given property p.

$$lsp = \uparrow_{\#} / \cdot (all\ p) \triangleleft \cdot segs$$

Property: lsp is not a homomorphism.

This is because:

$$\begin{array}{lll} lsp\ [2,1] & = lsp\ [2] & = [2] \\ lsp\ [4] & = lsp[4] & = [4] \end{array}$$

does not imply

$$lsp\ ([2,1] ++ [4]) = lsp\ ([2] ++ [4]).$$

$$
\begin{array}{ll}
& \uparrow_{\#} / \cdot (\mathit{all}\ p) \triangleleft \cdot \mathit{segs} \\
= & \quad \{ \text{ segment decomposition } \} \\
& \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot (\mathit{all}\ p) \triangleleft \cdot \mathit{tails}) * \cdot \mathit{inits} \\
= & \quad \{ \text{ result before } \} \\
& \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot (+\!\!+ / \cdot (p \rightarrow [id]^{o}, K_{\omega}) *) * \cdot \mathit{tails}) * \cdot \mathit{inits} \\
= & \quad \{ \text{ Horner's rule with } x \odot a = (x +\!\!+ (p\ a \rightarrow [a], \omega) \uparrow_{\#} [\,] \ \} \\
& \uparrow_{\#} / \cdot \odot \not\rightarrow_{[\,]} * \cdot \mathit{inits} \\
= & \quad \{ \text{ accumulation lemma } \} \\
& \uparrow_{\#} / \cdot \odot \#\!\!\!\!\rightarrow_{[\,]}
\end{array}
$$

Homework BMF 2-2

Show the final program for *lsp* is linear in the number of calculation of p, and code it in Haskell.